BUNDESREPUBLIK DEUTSCHLAND

Prioritätsbescheinigung über die Einreichung einer Patentanmeldung

Aktenzeichen:	103 14 835.3
Anmeldetag:	01. April 2003
Anmelder/Inhaber:	Siemens Aktiengesellschaft, 80333 München/DE
Bezeichnung:	Verfahren und Anordnung zur Erzeugung und Verarbeitung eines Quellcodes mit Spracherweiterungen
IPC:	G 06 F 9/45

Die angehefteten Stücke sind eine richtige und genaue Wiedergabe der ursprünglichen Unterlagen dieser Patentanmeldung.

München, den 15. April 2004
Deutsches Patent- und Markenamt
Der Präsident
Im Auftrag

Sieck

A 9161
06/00
EDV-L

Beschreibung

Verfahren und Anordnung zur Erzeugung und Verarbeitung eines Quellcodes mit Spracherweiterungen

Die Erfindung betrifft ein Verfahren und eine Anordnung zur Erzeugung und Verarbeitung von Quellcode, bei dem/der ein Quellcode in eine Darstellung in einer Meta-Auszeichnungssprache, beispielsweise XML, übergeführt, dort, beispielsweise mit XSLT, transformiert und dann diese in der Meta-Auszeichnungssprache formulierte transformierte Darstellung wieder in einen Quellcode umgewandelt wird.

Aus dem Internet ist unter http://beautyj.berlios.de/ ein Java Source Code Transformation Tool BeautyJ bekannt, bei dem ein Java Quellcode in eine XML-Darstellung umgewandelt wird, mittels Sourclet API, beispielsweise durch Einfügen von Leerzeichen oder geänderten Kommentaren an bestimmten Stellen, „verschönert" und anschließend der modifizierte Quellcode in Java Quellcode zurück konvertiert werden kann. Eine Transformation mittels XSLT wird hier, für diesen Zweck, nur vorgeschlagen, aber nicht umgesetzt.

Die der Erfindung zugrunde liegende Aufgabe liegt nun darin, ein Verfahren und eine Anordnung zur Erzeugung und Verarbeitung von Quellcode anzugeben, bei dem/der eine weitergehende noch flexiblere und effizientere Modifikation des Quellcodes erreicht wird.

Diese Aufgabe wird hinsichtlich des Verfahrens durch die Merkmale des Patentanspruchs 1 und hinsichtlich der Anordnung durch die Merkmale des Anspruchs 4 erfindungsgemäß gelöst. Die weiteren Ansprüche betreffen bevorzugte Ausgestaltungen der Erfindung.

Die Erfindung besteht im Wesentlichen darin, dass

ein in einer Meta-Auszeichnungssprache formulierter erster Code CodeML, der mindestens eine Spracherweiterung LE enthält und durch RCONV nicht in gültigen Quelltext SC* überführbar ist, durch eine Transformation T in Abhängigkeit von Transformationsregeln TR, welche einen Sprachkonverter LC enthalten, in einen in der Meta-Auszeichnungssprache formulierten zweiter Code CodeML* ohne diese Spracherweiterungen LE überführt wird, und dieser zweite Code CodeML* in einen in der ersten Programmiersprache ohne die Spracherweiterungen formulierten zweiten Quellcode SC* verwandelt wird. Ein wesentlicher Vorteil besteht bspw. darin, dass eine Verifikation oder Überprüfung der Spracherweiterungen mit einem für die normale Programmiersprache vorgesehenen Compiler erfolgen kann.

Die Erfindung wird im Folgenden anhand von in den Zeichnungen dargestellten Beispielen näher erläutert. Dabei zeigt

Zeichnung 1 ein Gesamtblockdiagramm zur Erläuterung der Erfindung und

Zeichnung 2 ein Blockschaltbild zur Erläuterung der erfindungsgemäßen Überführung von PreProcessing-Erweiterungen.

In **Zeichnung 1** ist ein Gesamtblockdiagramm zur Erläuterung der Erfindung dargestellt, bei dem ein in einer Meta-Auszeichnungsprache formulierter erster Code **CodeML**, der eine Spracherweiterung **LE** enthält und durch **RCONV** nicht in gültigen Quelltext **SC*** überführbar ist, durch eine Transformation **T** in Abhängigkeit von Transformationsregeln **TR**, welche einen Sprachkonverter **LC** enthalten, in einen in der Meta-Auszeichnungssprache formulierten zweiter Code **CodeML*** ohne überführt wird, welcher keine diese Spracherweiterungen **LE** enthält und deshalb in einen Quellcode **SC*** verwandelt werden kann, welcher mittels eines Compilers

COMP wiederum in gültigen Binärcode/ByteCode **B*** überführbar ist.

Der modifizierte Quellcode **SC*** sind beispielsweise in der Programmiersprache Java und die Codes **CodeML** und **CodeML*** sind beispielsweise in der Meta-Auszeichnungssprache XML formuliert, wobei „XML" für Extended Markup Language steht.

Die Transformation **T**, z. B. eine Extended Stylesheet Language Transformation oder **XSLT**, wird durch Transformationsregeln **TR**, z. B. innerhalb von **XSL** (Extended Stylesheet Language) Dateien beschrieben, wobei bspw. die in XSL formulierten Regeln u.a. als Language Converter LC verwendet werden und beschreiben wie der in XML-codierte Quellcode **CodeML** mit einer Spracherweiterung **LE** in eine Variante ohne Spracherweiterung transformierbar ist.

In **Zeichnung 2** ist ein erstes Ausführungsbeispiel dargestellt, bei dem ein in einer Meta-Auszeichnungssprache formulierter erster Code **CodeML** eine Spracherweiterung für PreProcessing **PPE** (z.B. <define>, <ifdef>, usw.) enthält, und mit Hilfe einer Transformtion **T** in Abhängigkeit von Transformationsregeln **TR**, welche einen PreProcessing Language Converter **PPLC** besitzen der **PPE** auflöst bzw. anwendet, in einen in der Meta-Auszeichnungssprache formulierten zweiten Code CodeML* ohne Spracherweiterung transformiert wird.

Die Ausgestaltung der Spracherweiterung erfolgt typischerweise in Form von Elementen für die generische Programmierung **1** , und/oder für PreProcessing **2** und/oder eine kunden- bzw. entwicklerspezifische Grammatik **3** und/oder für Makros **4**.

Alle Spracherweiterung bzw. der CodeML selbst kann jederzeit durch den Einsatz von DTDs(document type definitions) bzw. XMLSchema validiert werden.

Der Programmierer erhält durch die Erfindung mehr Freiheiten, da die Grammatik der verwendeten Programmiersprache auf die Wünsche des Programmierers abgestimmt werden kann und eine Rücktransformation auf die normale Grammatik der Programmiersprache erst am Ende der Programmentwicklung erfolgen muss. Ein wesentlicher Vorteil besteht auch darin, dass eine Validierung der Spracherweiterungen mit einem für die normale Programmiersprache vorgesehenen Compiler erfolgen kann.

Die im **Anhang 1** befindlichen Programmauflistungen **Listing 1** bis **Listing 3** zeigen die Auflösung der PreProcessing-Erweiterungen **PPE** an einem konkreten Beispiel, bei dem die in **Listing 1** dargestellte Klasse **TestOutput.xjava** eine **PPE** in Form von <define name="m" value="private"> enthält, welche Einfluß auf die Werte der <m>-Elemente nimmt, und durch eine Transformation **T** in Abhängigkeit von Transformationsregeln **TR** (hier: **PPLC**) nun in die in **Listing 2** dargestellte XML-basierte Form **TestOutput.xjava*** überführt wird, bei der alle <m>-Elemente durch ein über value="private" bestimmtes <private/>-Element ersetzt werden. Hiermit wird es möglich **TestOutput.xjava*** in den in **Listing 3** aufgezeigten Quellcode **TestOutput.java** zu überführen.

Durch das erfindungsgemäße Verfahren ergibt sich noch eine Reihe von weiteren Vorteilen, wie beispielsweise:

1. Es ist nur ein System für Problemstellungen wie Customizing von Programmiersprachen, usw. erforderlich und nicht eine Reihe verschiedener teilweise proprietärer Werkzeuge.

2. Das Verfahren basiert auf Standards wie XML und XSLT und ist hinsichtlich der Konvertierbarkeit in andere Programmiersprachen geringeren Beschränkungen unterworfen als andere Verfahren zur Modifikation von Quellcode.

3. Selbst für spezielle und komplizierte Quellcode-Modifikationen sind keine proprietären Speziallösungen erforderlich, sondern es können hierfür existierende Standards wie **XSLT, XPath** und **XQuery** genutzt werden.

4. Diese Art der Modifikation erlaubt die Erstellung von Hierarchien u.a. durch die Möglichkeit zur geordneten, automatisierten Hintereinanderausführung (Pipelines) mehrerer Transformationen, bspw. von Sprachanpassungen.

5. Die Transformationen können für eine allgemeine Wiederverwendung in XSLT-Dateien gespeichert werden, so daß Bibliotheken z.B. für bestimmte Abläufe entstehen können.

6. Es kann eine XML-Repräsentation des Quellcodes in einer XML-Datenbasis gespeichert und bei Bedarf mit Hilfe einer XSTL in einfacher Weise an die jeweiligen Kunden- bzw. Entwicklerbedürfnisse angepasst werden (Customization).

7. Durch die Verwendung der Standards **XMLSchema** oder **DTD** oder entsprechende XSLTs kann der Code vorab (ohne Kompilierung), auf bestimmte Korrektheitsaspekte hin, überprüft (validiert) werden.

8. Übliche XML-Tools können zur einfachen Bearbeitung bzw. Visualisierung und Bestimmung von Beziehungen im Code verwendet werden.

Anhang 1

Listing 1: TestOutput.xjava

```
<?xml version="1.0" encoding="UTF-8"?>
<java>
   <define name="m" value="private">
  <class>
    <modifiers><m/></modifiers>
    <name>TestOutput</name>
    <block>
      <var>
        <type><name>String</name></type>
        <name>m_sWelcome</name>
      </var>
    </block>
  </class>
</java>
```

Listing 2: TestOutput.xjava*

```
<?xml version="1.0" encoding="UTF-8"?>
<java>
  <class>
    <modifiers><private/></modifiers>
    <name>TestOutput</name>
    <block>
      <var>
        <type><name>String</name></type>
        <name>m_sWelcome</name>
      </var>
    </block>
  </class>
</java>
```

Listing 3: TestOutput.java

```
private class TestOutput
{
      String m_sWelcome;
}
```

Patentansprüche

1. Verfahren zur Erzeugung und Verarbeitung von Quellcode,
- bei dem ein in einer Meta-Auszeichnungsprache formulierter erster Code (CodeML) mit mindestens in einer Meta-Auszeichnungssprache formulierten Spracherweiterungen (LE) als Ausgangscode zur Verfügung steht,
- bei dem der Ausgangscode durch eine Transformation (T) in Abhängigkeit von Transformationsregeln (TR) in einen in der Meta-Auszeichnungssprache formulierter zweiter Code (CodeML*) ohne die in der Meta-Auszeichnungssprache formulierten Spracherweiterungen (LE) überführt wird,
- bei dem die Transformationsregeln einen Sprachkonverter (LC) enthalten, der die Spracherweiterungen (LE) des ersten Codes auflöst bzw. anwendet
- bei dem der erste Code nicht und der zweite Code prinzipiell durch einen Rückkonverter (RCONV) verarbeitet werden kann, und
- bei dem dieser zweite Code in einen in der ersten Programmiersprache oder einer anderen Programmiersprache formulierten zweiten Quellcode (SC*) umwandelbar ist, und einen gültigen Binärcode/ByteCode (B*) ergeben würde.

2. Verfahren nach Anspruch 1,
bei dem die Spracherweiterungen (LE) entweder PreProcessing-Erweiterungen (PPE), und/oder die generische Erweiterungen, und/oder kunden- bzw. entwicklerspezifische Erweiterungen und/oder für Makro-Erweiterungen enthält.

3. Verfahren nach Anspruch 1,
bei dem im zweiten Code (CodeML*) mindestens eine Spracherweiterung (LE*) neu erzeugt oder aus dem ersten Code (CodeML) übernommen wird, und diese Erzeugung oder Übernahme der Sprachkonverter (LC) vornimmt, und das Verfahren nach Anspruch 1 auf diesem zweiten Code erneut angewendet werden kann.

4. Verfahren nach einem der vorhergehenden Ansprüche, bei dem die Programmiersprache des zweiten Quellcodes Java und die Meta-Auszeichnungssprache XML ist und bei dem die Transformation und der Regelbeschreibung mittels XSLT und XSL erfolgt.

5. Anordnung zur Erzeugung und Verarbeitung von Quellcode,
- bei der ein Prozessor derart vorhanden ist, dass eine Transformation (T) des Ausgangscodes (CodeML) in Abhängigkeit von in einer erweiterten Stilbeschreibungssprache formulierten Transformationsregeln (TR) und einen darin enthaltenen Sprachkonverter (LC) so durchgeführt wird, dass entweder ein in der Meta-Auszeichnungssprache formulierter zweiter Code (CodeML*) ohne die in der Meta-Auszeichnungssprache formulierten Spracherweiterungen (LE) des ersten Codes (CodeML),
oder ein in der Meta-Auszeichnungssprache formulierter zweiter Code (CodeML*) mit neuen und/oder den ursprünglichen in der Meta-Auszeichnungssprache formulierten Spracherweiterungen (LE) erzeugt wird, und
- bei der ein Konverter (RCONV) derart vorhanden ist, dass dieser zweite Code in einen in der ersten Programmiersprache oder einer anderen Programmiersprache formulierten Quellcode (SC*) verwandelt wird.

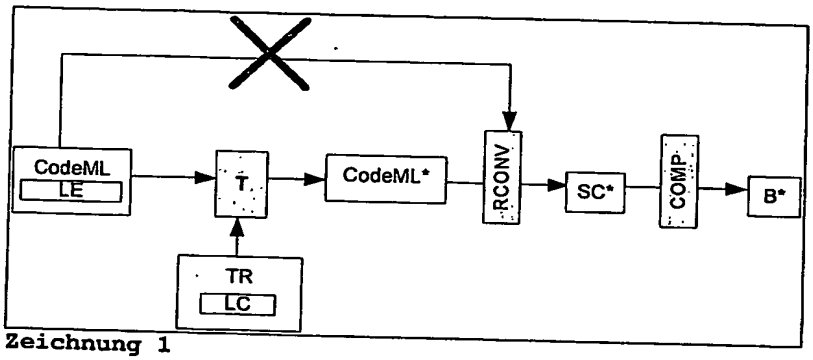
6. Anordnung nach Anspruch 5,
bei dem die Spracherweiterungen (LE) entweder als PreProcessing-Erweiterungen (PPE), und/oder als generische Erweiterungen, und/oder als kunden- bzw. entwicklerspezifische Erweiterungen und/oder als Makro-Erweiterungen vorhanden sind.

Zusammenfassung

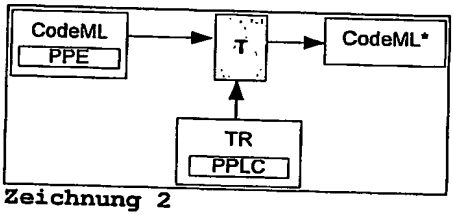
Verfahren und Anordnung zur Erzeugung und Verarbeitung eines Quellcodes mit Spracherweiterungen

Die Erfindung besteht im Wesentlichen darin, dass ein in einer Meta-Auszeichnungssprache formulierter erster Code CodeML, der mindestens eine Spracherweiterung LE enthält und durch RCONV nicht in gültigen Quelltext SC* überführbar ist, durch eine Transformation T in Abhängigkeit von Transformationsregeln TR, welche einen Sprachkonverter LC enthalten, in einen in der Meta-Auszeichnungssprache formulierten zweiter Code CodeML* ohne diese Spracherweiterungen LE überführt wird, und dieser zweite Code CodeML* in einen in der ersten Programmiersprache ohne die Spracherweiterungen formulierten zweiten Quellcode SC* verwandelt wird. Ein wesentlicher Vorteil besteht bspw. darin, dass eine Verifikation oder Überprüfung der Spracherweiterungen mit einem für die normale Programmiersprache vorgesehenen Compiler erfolgen kann.

Zeichnung 1



Zeichnung 1



Zeichnung 2